This listing of claims will replace all prior versions, and listings, of claims in the application.

LISTING OF CLAIMS:

Claim 1 (Currently Amended): A sampling-based system for adaptively optimizing a computer program executing in ~~an~~ a virtual machine execution environment, said virtual machine execution environment including one or more compiler devices for providing various levels of program optimization, said system comprising:

a runtime measurements sub-system for monitoring execution of said computer program to be optimized, said monitoring including obtaining raw profile data samples and characterizing said raw profile data;

a controller device for receiving said characterized raw profile data from said runtime measurements sub-system and analyzing said data for determining whether a predetermined level of program optimization for said executing program is to be performed by a compiler device, said controller generating a compilation plan in accordance with a determined level of optimization; and,

a recompilation sub-system for receiving a compilation plan from said controller and invoking a compiler device for performing said level of program optimization of said executing program in accordance with said compilation plan.

Claim 2 (Original): The system as claimed in Claim 1, wherein said runtime measurements sub-system comprises one or more organizer devices for processing said raw profile data and characterizing said data as meeting a hotness threshold of activity.

Claim 3 (Original): The system as claimed in Claim 2, wherein said runtime measurements sub-system comprises:

a mechanism for counting a number of samples that are taken from the executing program; and,

a mechanism for comparing the number of samples to a predetermined sampling size threshold, and in response to the number of samples exceeding said sampling threshold, invoking said organizer device to process said raw profile data.

Claim 4 (Original): The system as claimed in Claim 3, wherein said raw profile data samples relate to one or more method activations in said executing program.

Claim 5 (Original): The system as claimed in Claim 4, wherein said organizer device comprises a mechanism for comparing said raw profile data of method activations against a corresponding activity hotness threshold for one or more methods, and identifying one or more methods as meeting said activity hotness threshold for input to said controller device.

Claim 6 (Original): The system as claimed in Claim 3, wherein said controller device adaptively adjusts said sampling size threshold.

Claim 7 (Original): The system as claimed in Claim 4, wherein a level of program optimization includes recompiling an executing method, said controller device further comprising mechanism for adapting said sampling size threshold in accordance with an amount of recompilation that occurs.

Claim 8 (Original): The system as claimed in Claim 2, wherein said controller device dynamically adjusts said activity hotness threshold to adapt to a current behavior of the executing computer program.

Claim 9 (Original): The system as claimed in Claim 4, wherein said controller instructs the recompilation subsystem to insert intrusive profiling for one or more identified methods.

Claim 10 (Original): The system as claimed in Claim 4, wherein a compilation plan generated by said controller device comprises an identifier of a method to be optimized; and, an optimization level indicating a degree of optimization to be applied for said identified method.

Claim 11 (Original): The system as claimed in Claim 4, wherein a level of program optimization includes recompiling an executing method, said controller device including a mechanism for identifying a recompilation level that minimizes expected future running time of a recompiled program.

Claim 12 (Original): The system as claimed in Claim 11, wherein said mechanism for identifying a recompilation level includes:

mechanism for determining an expected time T_i the program will spend executing a method "m" if said method is not recompiled;

mechanism for determining a cost C_j of recompiling said method at an optimization level "j", for $i \leq j \leq N$; and,

mechanism for determining an expected time T_j the program will spend executing said method in the future, if said method is recompiled at level "j"; and,

comparison mechanism for evaluating the expression $C_j + T_j < T_i$, whereby said controller device decides to one of: generate compilation plan for directing recompilation of "m" at level "j" if said expression is true, and, not recompile if said expression is false.

Claim 13 (Original): The system as claimed in Claim 1, wherein said raw profile data samples are taken at method prologue and back edge yield points.

Claim 14 (Original): The system as claimed in Claim 1, wherein said controller instructs the recompilation subsystem to perform online feedback-directed optimizations based on feedback from the current executing program.

Claim 15 (Original): The system as claimed in Claim 14, wherein said raw profile data samples relate to call context information associated with methods called by said program, said feedback comprising said call context information.

Claim 16 (Original): The system as claimed in Claim 14, wherein said raw profile data samples relate to current program variable values, said feedback comprising a subset of values assigned to said variables during program execution.

Claim 17 (Original): The system as claimed in Claim 14, wherein said raw profile data samples relate to control flow execution within a method, said feedback comprising execution frequency of control flow paths within said executing method.

Claim 18 (Original): The system as claimed in Claim 1, wherein said execution environment includes an interpreter device.

Claim 19 (Currently Amended): A method for adaptively optimizing a computer program executing in ~~an~~ a virtual machine execution environment, said virtual machine execution environment comprising one or more compiler devices for providing various levels of program optimization, said method comprising:

a) sampling said executing computer program to obtain raw profile data samples;

b) characterizing said raw profile data as meeting a threshold criteria;

c) analyzing said characterized raw profile data for determining whether a predetermined level of program optimization for said executing program is to be performed by a compiler device, and generating a compilation plan in accordance with a determined level of optimization; and,

d) when optimization is to be performed, invoking a compiler device for optimizing said executing program in accordance with said compilation plan.

Claim 20 (Original): The method as claimed in Claim 19, wherein said characterizing step b) comprises: processing said raw profile data to determine whether said data meets an activity hotness threshold.

Claim 21 (Original): The method as claimed in Claim 20, wherein said sampling includes the steps of:

counting said samples that are taken from the executing program; and,

comparing the amount of samples taken to a predetermined sampling size threshold, whereby in response to an amount of samples exceeding said sampling threshold, performing said characterizing step.

Claim 22 (Original): The method as claimed in Claim 21, wherein said raw profile data sampled relates to one or more method activations in said executing program.

Claim 23 (Original): The method as claimed in Claim 22, wherein said step of processing said raw profile data comprises the steps of:

comparing said raw profile data of method activations against said corresponding activity hotness threshold for one or more methods; and

identifying one or more methods as meeting said activity hotness threshold.

Claim 24 (Original): The method as claimed in Claim 21, further including the step of adaptively adjusting said sampling size threshold.

Claim 25 (Original): The method as claimed in Claim 22, wherein said step of optimizing includes recompiling an executing method meeting an activity hotness threshold, said method further including the step of adapting said sampling size threshold in accordance with an amount of recompilation that occurs.

Claim 26 (Original): The method as claimed in Claim 22, further including the step of dynamically adjusting said activity hotness threshold for adapting to a current behavior of the executing computer program.

Claim 27 (Original): The method as claimed in Claim 26, wherein said step of dynamically adjusting said activity hotness threshold further includes the steps of:

determining an amount of methods characterized as meeting said threshold criteria after one or more sampling periods;

comparing said amount to a limit; and,

in response to said comparing, one of: decreasing said activity hotness threshold if said limit is not met, and increasing the threshold if said limit is met or exceeded in said one or more sampling periods.

Claim 28 (Original): The method as claimed in Claim 22, further including the step of inserting intrusive profiling for one or more identified methods.

Claim 29 (Original): The method as claimed in Claim 24, wherein said step of generating a compilation plan includes: providing an identifier of said method to be optimized; and, an optimization level indicating a degree of optimization to be applied for said identified method.

Claim 30 (Original): The method as claimed in Claim 22, wherein said analyzing step c) further includes identifying a recompilation level that minimizes expected future running time of a recompiled version.

Claim 31 (Original): The method as claimed in Claim 30, wherein said step of identifying a recompilation level includes the steps of:

determining an expected time T_i the program will spend executing a method "m" if said method is not recompiled;

determining a cost C_j of recompiling said method at an optimization level "j", for $i \leq j \leq N$; and,

determining an expected time T_j the program will spend executing said method in the future, if said method is recompiled at level "j"; and,

evaluating the expression $C_j + T_j < T_i$, and, one of: recompiling "m" at level "j" if said expression is true, and, not recompiling if said expression is false.

Claim 32 (Original): The method as claimed in Claim 19, wherein said optimizing step further comprises the step of performing online feedback-directed optimizations based on feedback from the current executing program.

Claim 33 (Original): The method as claimed in Claim 32, wherein said raw profile data samples relate to call context information associated with methods called by said program, said feedback comprising said call context information.

Claim 34 (Original): The method as claimed in Claim 32, wherein said raw profile data samples relate to current program variable values, said feedback comprising a subset of values assigned to said variables during program execution.

Claim 35 (Original): The method as claimed in Claim 32, wherein said raw profile data samples relate to control flow execution within a method, said feedback comprising execution frequency of control flow paths within said executing method.

Claim 36 (Original): The method as claimed in Claim 19, wherein said execution environment includes an interpreter device.

Claim 37 (Currently Amended): A computer program product comprising a computer readable medium having recorded thereon a computer program which, when loaded in a computer, configures a computer for adaptively optimizing a computer program executing in ~~an~~ a virtual machine execution environment, said virtual machine execution environment comprising one or more compiler devices for providing various levels of program optimization, said computer program executing method steps comprising:

a) sampling said executing computer program to obtain raw profile data samples;

b) characterizing said raw profile data as meeting a threshold criteria;

c) analyzing said characterized raw profile data for determining whether a predetermined level of program optimization for said executing program is to be performed by a compiler device, and generating a compilation plan in accordance with a determined level of optimization; and,

d) when optimization is to be performed, invoking a compiler device for optimizing said executing program in accordance with said compilation plan.

Claim 38 (Original): The computer program product as claimed in Claim 37, wherein said characterizing step b) comprises: processing said raw profile data to determine whether said data meets an activity hotness threshold.

Claim 39 (Original): The computer program product as claimed in Claim 37, wherein said sampling step includes the steps of:

counting said samples that are taken from the executing program; and,

comparing the amount of samples taken to a predetermined sampling size threshold, whereby in response to an amount of samples exceeding said sampling threshold, performing said characterizing step.

Claim 40 (Original): The computer program product as claimed in Claim 39, wherein said raw profile data sampled relates to one or more method activations in said executing program.

Claim 41 (Original): The computer program product as claimed in Claim 40, wherein said step of processing said raw profile data comprises the steps of:

comparing said raw profile data of method activations against said corresponding activity hotness threshold for one or more methods; and

identifying one or more methods as meeting said activity hotness threshold.

Claim 42 (Original): The computer program product as claimed in Claim 39, further including the step of adaptively adjusting said sampling size threshold.

Claim 43 (Original): The computer program product as claimed in Claim 40, wherein said step of optimizing includes recompiling an executing method meeting an activity hotness threshold, said method further including the step of adapting said sampling size threshold in accordance with an amount of recompilation that occurs.

Claim 44 (Original): The computer program product as claimed in Claim 40, further including the step of dynamically adjusting said activity hotness threshold for adapting to a current behavior of the executing computer program.

Claim 45 (Original): The computer program product as claimed in Claim 44, wherein said step of dynamically adjusting said activity hotness threshold further includes the steps of:

determining an amount of methods characterized as meeting said threshold criteria after one or more sampling periods;

comparing said amount to a limit; and,

in response to said comparing, one of: decreasing said activity hotness threshold if said limit is not met, and increasing the threshold if said limit is met or exceeded in said one or more sampling periods.

Claim 46 (Original): The computer program product as claimed in Claim 40, further including the step of inserting intrusive profiling for one or more identified methods.

Claim 47 (Original): The computer program product as claimed in Claim 42, wherein said step of generating a compilation plan includes: providing an identifier of said method to be optimized; and, an optimization level indicating a degree of optimization to be applied for said identified method.

Claim 48 (Original): The computer program product as claimed in Claim 40, wherein said analyzing step c) further includes identifying a recompilation level that minimizes expected future running time of a recompiled version.

Claim 49 (Original): The computer program product as claimed in Claim 48, wherein said step of identifying a recompilation level includes the steps of:

determining an expected time T_i the program will spend executing a method "m" if said method is not recompiled;

determining a cost C_j of recompiling said method at an optimization level "j", for $i \leq j \leq N$; and,

determining an expected time T_j the program will spend executing said method in the future, if said method is recompiled at level "j"; and,

evaluating the expression $C_j + T_j < T_i$, and, one of: recompiling "m" at level "j" if said expression is true, and, not recompiling if said expression is false.

Claim 50 (Original): The computer program product as claimed in Claim 37, wherein said optimizing step further comprises the step of performing online feedback-directed optimizations based on feedback from the current executing program.

Claim 51 (Original): The computer program product as claimed in Claim 50, wherein said raw profile data samples relate to call context information associated with methods called by said program, said feedback comprising said call context information.

Claim 52 (Original): The computer program product as claimed in Claim 50, wherein said raw profile data samples relate to current program variable values, said feedback comprising a subset of values assigned to said variables during program execution.

Claim 53 (Original): The computer program product as claimed in Claim 50, wherein said raw profile data samples relate to control flow execution within a method, said feedback comprising execution frequency of control flow paths within said executing method.

Claim 54. (Original) The computer program product as claimed in Claim 37, wherein said execution environment includes an interpreter device.